



04.10 - OSCO

03CO (7)

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant : Alistair Goudie
Serial No.: 10/035,930
Filed: December 26, 2001
Title: ARBITER FOR A QUEUE MANAGEMENT SYSTEM
Docket No.: 34260

LETTER

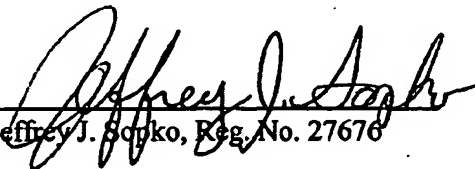
Asst. Commissioner of Patents
Washington, D.C. 20231

Sir/Madam:

Enclosed is a certified copy of British Patent Application No. 0031763.6; the
priority of which has been claimed in the above-identified application.

Respectfully submitted,

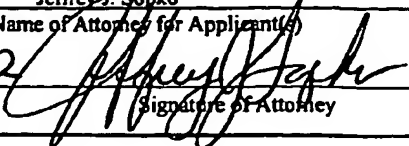
PEARNE & GORDON, LLP

By: 
Jeffrey J. Sopko, Reg. No. 27676

526 Superior Avenue, East
Suite 1200
Cleveland, Ohio 44114-1484
(216) 579-1700

Date: 3/21/02

I hereby certify that this correspondence is being deposited
with the United States Postal Service as first class mail in an
envelope addressed to: Assistant Commissioner for Patents,
Washington D.C. 20231 on the date indicated below.

Jeffrey J. Sopko
Name of Attorney for Applicant(s)
3/21/02 
Date Signature of Attorney

This Page Blank (uspto,



INVESTOR IN PEOPLE

The Patent Office
Concept House
Cardiff Road
Newport
South Wales
NP10 8QQ

the undersigned, being an officer duly authorised in accordance with Section 74(1) and (4) of the Deregulation & Contracting Out Act 1994, to sign and issue certificates on behalf of the Comptroller-General, hereby certify that annexed hereto is a true copy of the documents as originally filed in connection with the patent application identified therein.

In accordance with the Patents (Companies Re-registration) Rules 1982, if a company named in this certificate and any accompanying documents has re-registered under the Companies Act 1980 with the same name as that with which it was registered immediately before re-registration, whether by the substitution as, or inclusion as, the last part of the name of the words "public limited company" or their equivalents in Welsh, references to the name of the company in this certificate and any accompanying documents shall be treated as references to the name with which it is so registered.

In accordance with the rules, the words "public limited company" may be replaced by p.l.c., plc, P. L. C. or PLC.

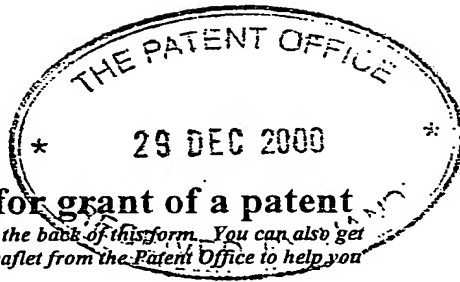
Re-registration under the Companies Act does not constitute a new legal entity but merely subjects the company to certain additional company law rules.

**CERTIFIED COPY OF
PRIORITY DOCUMENT**

Signed *AmBrewer*

Dated 29 January 2002

This Page Blank (uspto)



The
**Patent
Office**

31DEC00 E594624-5 001038
P01/7700 0.00-0031763.6

1/77

Request for grant of a patent

(See the notes on the back of this form. You can also get an explanatory leaflet from the Patent Office to help you fill in this form)

The Patent Office
Cardiff Road
Newport
South Wales
NP10 8QQ

1. Your reference P502838GB/46100

2. Patent application number
(The Patent Office will fill in this part)

0031763.6

3. Full name, address and postcode of the or of each applicant (underline all surnames)

Mitel Semiconductor Limited
Cheney Manor
Swindon
Wiltshire
SN2 2QW
United Kingdom

Patents ADP number (if you know it)

7387442001

If the applicant is a corporate body, give the country/state of its incorporation

United Kingdom

4. Title of the invention

Arbiter for a Queue Management System

5. Name of your agent (if you have one)

"Address for service" in the United Kingdom to which all correspondence should be sent (including the postcode)

WITHERS & ROGERS
Goldings House
2 Hays Lane
London
SE1 2HW

Patents ADP number (if you know it)

1776001

6. If you are declaring priority from one or more earlier patent applications, give the country and the date of filing of the or each of these earlier applications and (if you know it) the or each application number

Country

Priority application number
(if you know it)

Date of filing
(day / month / year)

7. If this application is divided or otherwise derived from an earlier UK application, give the number and the filing date of the earlier application

Number of earlier application

Date of filing
(day / month / year)

8. Is a statement of inventorship and of right to grant of a patent required in support of this request? (answer 'Yes' if:

- a) any applicant named in part 3 is not an inventor, or
b) there is an inventor who is not named as an applicant, YES
or
c) any named applicant is a corporate body.
See note (d))

Patents Form 1/77

9. Enter the number of sheets for any of the following items you are filing with this form.
Do not count copies of the same document

Continuation sheets of this form

Description 9

Claim(s) 1

Abstract 1

Drawing (s) 3 + 3

10. If you are also filing any of the following, state how many against each item.

Priority documents

Translations of priority documents

Statement of inventorship and right to grant of a patent (Patents Form 7/77)

Request for preliminary examination and search (Patents Form 9/77) 1

Request for substantive examination (Patents Form 10/77) 1

Any other documents (please specify)

11. I/We request the grant of a patent on the basis of this application.

W. Withers & Rogers

Signature Withers & Rogers

Date 29 December 2000

12. Name and daytime telephone number of person to contact in the United Kingdom

David M Pratt

020 7663 3500

Warning

After an application for a patent has been filed, the Comptroller of the Patent Office will consider whether publication or communication of the invention should be prohibited or restricted under Section 22 of the Patents Act 1977. You will be informed if it is necessary to prohibit or restrict your invention in this way. Furthermore, if you live in the United Kingdom, Section 23 of the Patents Act 1977 stops you from applying for a patent abroad without first getting written permission from the Patent Office unless an application has been filed at least six weeks beforehand in the United Kingdom for a patent for the same invention and either no direction prohibiting publication or communication has been given, or any such direction has been revoked.

Notes

- If you need help to fill in this form or you have any questions, please contact the Patent Office on 0645 500 505.
- Write your answers in capital letters using black ink or you may type them.
- If there is not enough space for all the relevant details on any part of this form, please continue on a separate sheet of paper and write "see continuation sheet" in the relevant part(s). Any continuation sheet should be attached to this form.
- If you have answered 'Yes' Patents Form 7/77 will need to be filed.
- Once you have filled in the form you must remember to sign and date it.
- For details of the fee and ways to pay please contact the Patent Office.

Arbiter for a Queue Management System

This invention relates to an arbiter for a system containing multiple bus masters, , and in particular to a bus arbiter in a queue management system (QMS) block in a Bluetooth baseband peripheral.

A bus arbiter for a QMS is used to decide bus mastership in a system having multiple masters each having real time requirements. A master with real time requirements is not likely to need to access the bus all the time, but it does need to be sure it can make a predetermined number of bus accesses within a given period. Known arbiter blocks have a fixed set of priority levels so that, if a high priority block requests mastership, it prevents bus access to all lower priority blocks. In some known systems, all the prioritisation levels are required to be shuffled, but this rarely leads to a situation that can guarantee bandwidth to multiple arbiter blocks.

15

The present invention provides an arbiter for a system having multiple bus masters each having real time requirements for mastership of a bus, wherein the arbiter is arranged so that the amount of time that each bus master can gain bus access is a percentage of the total bus time.

20

This ensures worse case conditions for bus access can be clearly defined, and will not be affected by adding or changing the functions of other bus masters. Consequently, bus access is evenly distributed, rather than occurring in bursts.

25 In a preferred embodiment, the system comprises a QMS, and each bus master is a queue user. In this case, the QMS may include a queue portal for each of the queue users, a respective queue user interface being positioned between each queue user and its portal.

30 Advantageously, the arbiter includes a state machine for allocating each bus master its predetermined percentage of bus time. Preferably, the state machine determines the

active queue portal arbitration by cycling through a predetermined number of states, in a fixed order, or every clock cycle, each of the states being associated with a respective queue portal. Thus, by allocating appropriate numbers of states to the queue users, each can be allocated its percentage of total bus time.

5

The invention will now be described in greater detail, by way of example, with reference to the drawings in which:-

Figure 1 shows the system architecture for a QMS;

Figure 2 shows the system architecture for a QMS which is used for a Bluetooth
10 baseband peripheral; and

Figure 3 is a flow diagram illustrating the decision processes of a state machine forming part of the architecture of Figure 2.

Referring to the drawings, Figure 1 shows a system architecture which includes a QMS,
15 indicated by the dotted-line block 1, a queue manager software interface 2, a core (queue manager hardware) 3, and four identical queue portals 4A, 4B, 4C and 4D. The portals 4A to 4D are associated with respective queue users 5A to 5D by means of queue interfaces 6A to 6D. The queue users 5A to 5D are external entities that accesses data stored in a queue. The core 3 includes an arbiter 7, a "memory
20 operation" block 8, and a re-allocation block 9. The block 9 includes a block release table 9a and a removed blocks FIFO 9b, and communicates with a single port RAM 10. The arbiter 7 controls access to a bus 11 in a manner described below.

The QMS 1 uses the RAM 10 to hold data as it passes through the queues. The QMS 1
25 can support two types of queues, namely asynchronous queues (which are intended to hold 'good' data which can be held for an indeterminate amount of time), and isochronous queues (which are intended to hold a small amount of time-critical data). The mean data rate of the generator and consumer of data in an isochronous queue should be approximately the same. The data may be read/written in bursts, which is
30 typically why the queue system is required. For isochronous queues, the QMS 1 splits the RAM 10 into memory blocks, which are linked together to form variable length FIFO queues. The QMS 1 can resize a queue (by adjusting the rules for adding memory

blocks) as the storage requirement changes. For isochronous queues, a smaller area of contiguous memory is used.

As mentioned above, for each queue user 5A to 5D there is a dedicated generic queue portal 4A to 4D. A queue portal 4A to 4D can be used to access a number of queues, although it can only read/write to one queue at a time. A queue user 5A to 5D will indicate which queues it wants to access, and will then do a series of reads (or writes). The queue portal 4A to 4D will map the reads (or writes) into RAM accesses. Within a memory block, the data will be stored at sequential locations. At the end of a memory block, the relevant queue portal 4A to 4D will follow a link to the next memory block in a chain, and start accessing the new memory block. The queue user 5A to 5D will not be aware of the linking process (although it will introduce delays to some accesses).

The read (or write) operations automatically update memory pointers provided in the queue portals 4A to 4D. However, for asynchronous queues in which data is expected to be good, there may be a requirement to repeat a series of reads/writes (e.g. due to a requirement to re-transmit a packet, or due to an error in a received packet). Therefore, the queue portals 4A to 4D each has a commit/discard mechanism. As well as the current/write pointer (which is updated every read/write), there is a second pointer which can identify the start of a data block. A "data block" is not the same thing as a memory block, being a series of consecutive reads/writes, which may be stored within a memory block, or in a number of memory blocks. If a queue user 5A to 5D decides a data block should be re-read/re-written, it can do a 'discard', which loads the last stored 'start of data block' pointer into the current pointer. If the queue user 5A to 5D decides a data block is good, it can do a 'commit' which loads the current pointer into the 'start of data block' pointer. When a queue user 5A to 5D is writing to a queue, it will always decide to commit/discard before unloading the queue.

The QMS 1 also includes four flag bits that can indicate boundaries in the data flow. The flags are stored at the start of a memory block (they cannot identify a location within the block). If the most-significant bit (MSB) of the flags is set, the queue portal 4A to 4D will advance the current pointer to the start of the next memory block. New

data will be stored in the new memory block, rather than filling the previous block first. The other flags will be used to identify boundaries over multiple MSB boundaries. The flags are not necessarily used at the same time as the commit/discard mechanism. The queue user 5A to 5D could decide to write several packets before 'committing' to them, or it could 'commit' to a fragment of a packet. The flags can be used to help with fragmentation and re-assembly of packets, and/or to identify packet headers stored with the data.

For isochronous queues, a simpler mechanism is required. The commit/discard mechanism and start of packet indication are not required. The isochronous queues are intended to hold error tolerant, time-critical data (e.g. voice samples). The queue uses a simple FIFO approach (with pointers mapped to an area of the single port RAM 10). If the data generator fills the FIFO, it will advance both the generator and consumer pointers, so the oldest data in the FIFO is discarded. If the data consumer empties the FIFO, an error will be indicated, and the last data read from the FIFO will be repeated.

Figure 2 shows the system architecture of a Bluetooth baseband peripheral (indicated by the dotted line B) which contains the hardware required to implement a Bluetooth baseband control. The peripheral is designed to work as a slave peripheral device that can interface to a 16-bit processor or a 32-bit processor (the block uses a 16-bit address and a 16-bit data bus). The peripheral block includes link control hardware required to communicate with other Bluetooth devices via a radio IC and a QMS 21. As with the embodiment of Figure 1, the QMS 21 is controlled via a queue manager software interface (not shown). The QMS 21 is identical to the QMS 1 of Figure 1, and so is not shown in detail in Figure 2. Thus, the QMS 21 communicates with a single port RAM 30, and controls access to a bus 31. The bus 31 is associated with a "Firefly" block 32, which is a re-usable microprocessor block which consists of an ARM7TDMI processor and ancillary blocks including a memory/peripheral controller, and an interrupt controller and a UART.

The QMS 21 decides bus mastership for queue users 25A to 25D (only 25A to 25C of which are shown - the processor 25D being constituted by the ARM7TDMI processor in the Firefly block 32), each of which has an associated user interface 26A to 26D. The queue user 25A is a communication control block (CCB), the queue user 25B is a host queue user (HOSTIF), the queue user 25C is a voice encoder and decoder (CODEC), and the queue user 25D is a processor. The queue portals associated with the interfaces 26A to 26D are not shown in Figure 2, being part of the QMS 21.

The QMS 21 communicates with the bus 31 via a bus master 33A and a bus tri-state driver 33B. Similarly, the user interfaces 26A to 26D are connected to the bus 31 via respective bus masters 34A, 35A, 36A and 37A and respective bus tri-state drivers 34B, 35B, 36B and 37B. The queue user 25A is connected to the bus 31 by a bus slave 38, and the CODEC 25C is connected to the bus 31 by a bus slave 39. The bus 31 is connected to the Firefly block 32 via up-integration module (UIM) to bus interface (UBI) block 40 by a UIM bus 41. The processor user interface 26D is connected to the block 40 by a direct memory access (DMA) upload/download connection 42. The block 40 isolates the UIM bus 41 from the bus 31. The processor accesses the peripheral block B by getting the UBI block 40 to request bus mastership.

The peripheral block B thus has three main parts, namely:-

1. A link controller that interfaces to a radio, the link controller comprising the CCB 25A, the CCB queue user interface 26A, a CCB radio interface (CRI) 25R and a voice encoding translator (VET) 26T.
2. A buffer manager that stores data packets and allows processor interaction, the buffer manager comprising the UBI 40, the processor interface 26D, the QMS 21, and a block of the single port RAM 30.
3. A host interface that interfaces to an external host device (not shown), the host interface comprising the host queue user interface 26B, the host interface 25B (which may be a UART), the voice queue user interface 26C and the CODEC 25C.

The arbiter 7 of the QMS 21 is responsible for determining which queue user 25A to 25D can access the bus 31, which in turn effects which user can access the RAM 32. An external 'arbit_en' input signal (see Figure 1) acts as an enable to the arbiter 7.

- 5 When enabled, the arbiter 7 includes the following priority levels.

Priority	Function
Highest	Continuation of un-interruptible operation
....	Active queue portal (round robin approach to provide guaranteed bandwidth)
....	Processor initiated start of un-interruptible operation
....	Processor access to Bluetooth block
Lowest	Re-allocation function

- 10 A state machine constituted by the arbiter 7 determines the 'active queue portal' arbitration. Every clock cycle, the state machine 7 cycles through the states, in a fixed order. The only exception is non-interruptible sequences which make the state machine wait in its current state. This sequence is:-

State	Priority Queue Portal
0	The queue portal for the user 25D
1	The queue portal for the user 25B
2	The queue portal for the user 25D
3	The queue portal for the user 25B
4	The queue portal for the user 25D
5	The queue portal for the user 25B
6	The queue portal for the user 25A
7	The queue portal for the user 25B
8	The queue portal for the user 25D
9	The queue portal for the user 25B
10	The queue portal for the user 25C
11	The queue portal for the user 25B
12	The queue portal for the user 25D
13	The queue portal for the user 25B
14	The queue portal for the user 25A
15	The queue portal for the user 25B

This system ensures a certain percentage of the bus bandwidth is available for each queue portal. The percentage of bus bandwidth allocated should be sufficient to meet the burst requirements of the queue portals, the HOSTIF 25B being allocated 50%, the processor 25D being allocated 31.25%, the CCB 25A being allocated 12.5%, and the CODEC 25C being allocated 6.25%. The queue portals should work with less than the allocated bandwidth, due to non-interruptible sequences that can temporarily increase the bandwidth used by other portals. (The implementation should allow these allocations to be changed easily).

10

The arbiter 7 also includes an enable bit for each queue portal. The enable bit can be used to block requests for mastership (although this will not effect the sequence of the state machine).

15 If the queue portal currently selected by the state machine 7 does not request bus mastership, the one of the lower priority functions can become bus master. If the processor 25D triggers a non-interruptible memory sequence, it has a higher priority than standard processor accesses (this ensures the sequence will be complete before the processor can read back the result). The processor is the next highest priority master.

20 If a high priority queue portal or a non-interruptible sequence is active, the processor is held in a wait state. When the higher priority masters are no longer using the bus 31, the processor 25D is granted bus mastership. If both the high priority queue portal and the processor 25D do not request the bus 31, the QMS 21 can use the 'spare' bus cycles to do low priority accesses, such as block re-allocation.

25

Figure 3 is a flow chart illustrating the decision processes of the state machine 7. The processes start at step 100, and, in step 101 the arbiter 7 checks to see if it is enabled. If not, the block 40 has bus mastership (step 102), and the state machine proceeds to step 103 to wait for the next clock cycle, at which stage step 101 is repeated.

30

If the arbiter 7 is enabled, step 104 queries whether the current bus master requests retention of mastership. If so, that bus master is granted retention of bus mastership (in

step 105), and the state machine then returns to step 103 to wait for the next clock cycle. If the current bus master did not request retention of mastership, the state machine proceeds to step 106, where the queue portal is advanced in round robin fashion. If the queue portal selected in this manner requests bus mastership step (107),

5 the selected queue portal is granted bus mastership (or passes mastership to the associated queue user) - in step 108. The programme then returns to step 103 to wait for the next clock cycle.

If the queue portal selected does not request bus mastership, the QMS core is asked

10 whether a processor initiated memory operation has requested bus mastership (step 109). If the answer to this question is yes, a memory operation block in the QMS core is granted bus mastership (step 110), and the state machine then returns to step 103 to wait for the next clock cycle. If a processor initiated memory operation has not requested bus mastership, a check is made (step 111) to see whether the processor 25D

15 has requested a bus mastership via the block 40. If such a request has been made, the block 40 is granted bus mastership (step 112) and the state machine returns to step 103 to wait for the next clock cycle.

If the processor 25D did not request bus mastership via the block 40, a check is then

20 made (step 113) to see whether the re-allocation block 9 has requested bus mastership. If it has, a check is made to see whether there has been a block release table operation (step 114). If a block release table operation is required, this is carried out in step 115; and, if not, the blocks are removed from the FIFO 9b (step 116). In either case, the state machine then returns to step 103 to wait for the next clock cycle. If the

25 re-allocation block 9 has not requested bus mastership the QMS is given bus mastership (step 117), after which the state machine returns to step 103 to wait for the next clock cycle. In step 117, at all times there must be one (and only one) bus master. Hence, if no other block wants to be bus master, the QMS block 21 must be the bus master. However, the QMS 21 doesn't want to have anything to do with the bus 31, so it drives

30 the address and control signals of the bus into a state where it is not actively reading/writing to any blocks.

It will be apparent that modifications would be made to the arbiter described above. Thus, the version of the arbiter described above includes a 'hold' signal to allow a block to do an un-interruptible read-modify-write cycle. Unfortunately, this re-introduces some uncertainty into the number of bus cycles allocated (because the 'hold' cycles are not counted). If this becomes a problem, the arbiter could take account of 'hold' cycles when allocating future bus access (e.g. if a block introduces a hold cycle, the next time the arbiter is going to allocate bus mastership to the block, it could skip it).

The arbiter described above uses a state machine with sixteen states. If a finer solution in the allocation of bus mastership is required, either more states could be introduced, or a numerical algorithm could be used.

The arbiter described above could be used with any system that needs to control access to a resource (usually a bus). Its greatest strength is defining the percentage of time that a unit can access a resource. This gives much more certainty to defining the design requirements of a block accessing the resource, and any given block doesn't have to worry about that other blocks accessing the resource do. This is to be contrasted with known systems with a prioritised access, where changes to a high priority block could affect the behaviour of a lower priority block).

Claims

1. An arbiter for a system having multiple bus masters each having real time requirements for mastership of a bus, wherein the arbiter is arranged so that the amount of time that each bus master can gain bus access is a percentage of the total bus time.
5
2. An arbiter as claimed in claim 1, wherein the system comprises a QMS, and each bus master is a queue user.
- 10 3. An arbiter as claimed in claim 2, wherein the QMS includes a queue portal for each of the queue users, a respective queue user interface being positioned between each queue user and its portal.
4. An arbiter as claimed in any one of claims 1 to 3, wherein the arbiter includes a state machine for allocating each bus master its predetermined percentage of bus time.
15
5. An arbiter as claimed in claim 4 when appendant to claim 3, wherein the state machine determines the active queue portal arbitration by cycling through a predetermined number of states, in a fixed order, or every clock cycle, each of the states
20 being associated with a respective queue portal.

Arbiter for a Queue Management System**Abstract**

5 An arbiter (7) is provided for a QMS having multiple queue users (5A to 5D), each having real time requirements for mastership of a bus (31). The arbiter (7) is arranged so that the amount of time that each queue user (5A to 5D) can gain bus access is a percentage of the total bus time.

THIS PAGE BLANK (USPTO)

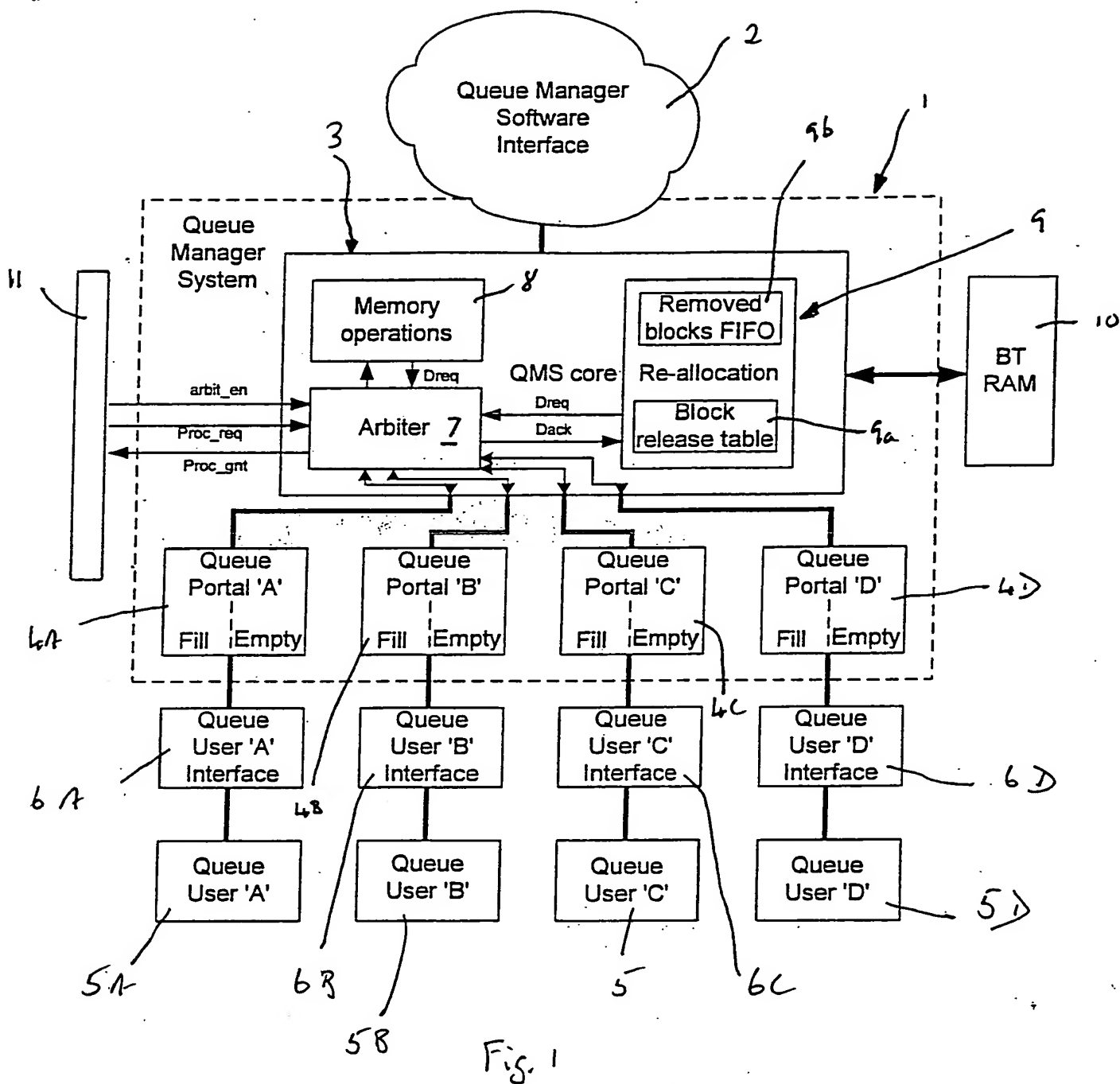


Fig. 1

THIS PAGE BLANK (USPTO)

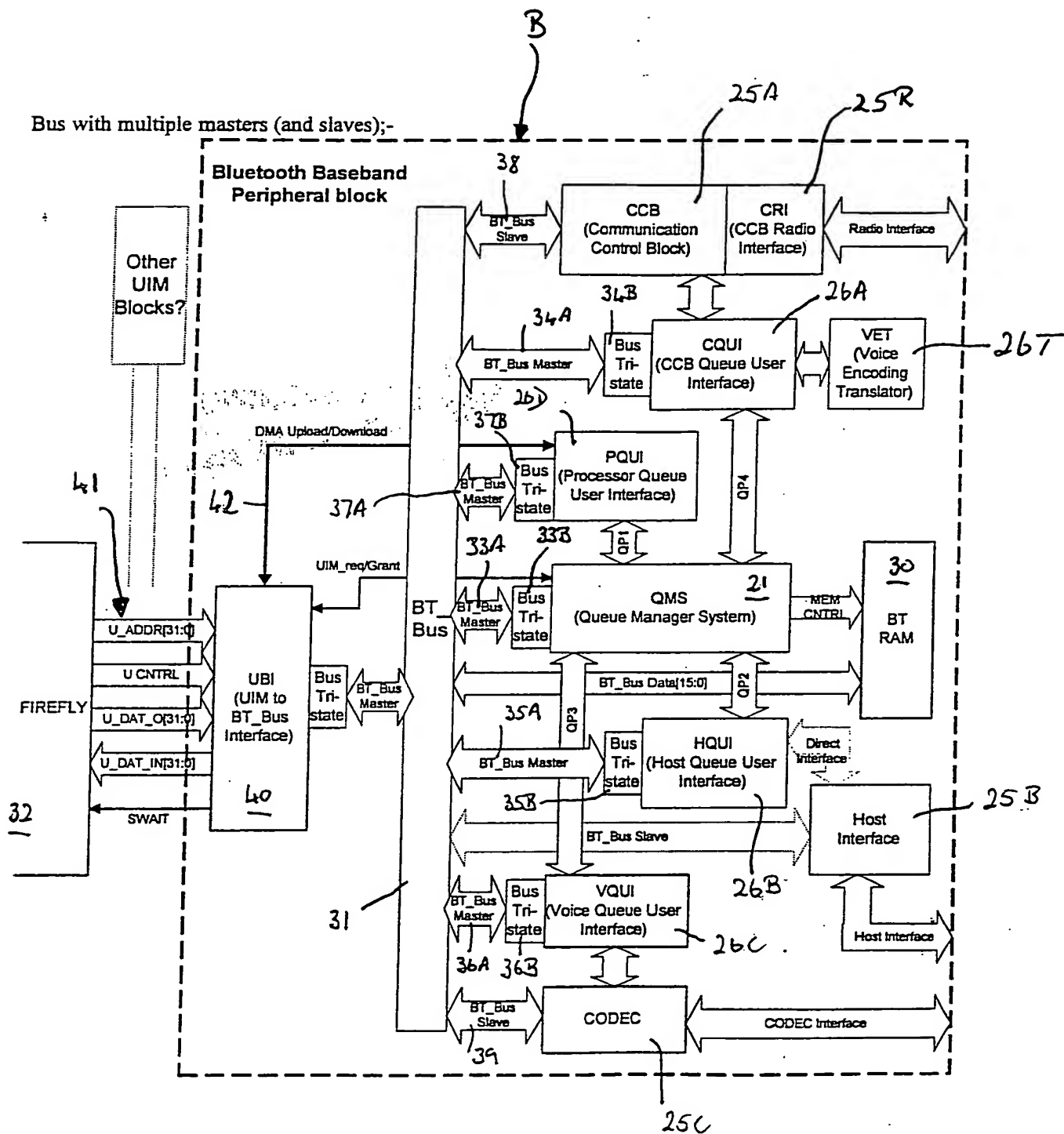


Fig. 2

THIS PAGE BLANK (USPTO)

Arbiter state machine;-

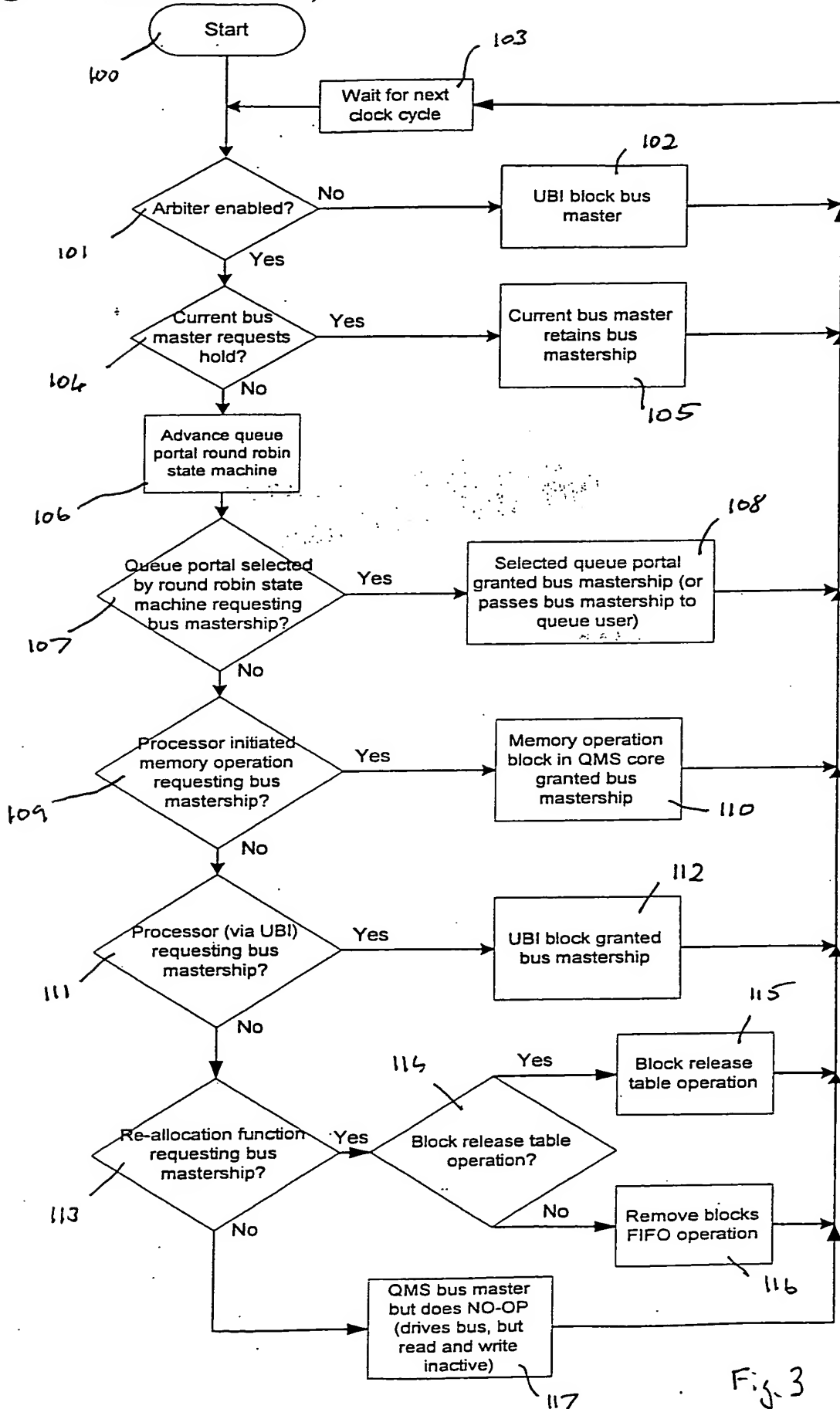


Fig. 3

This Page Blank (uspto)